points-frame: when your sensor is your source of coordinates

Assume that you happened to know the coordinates of your sensor in some Cartesian coordinate system, and you want to derive the coordinates of your robot centre. For the robot configuration you know the offset of the sensor *from* the robot centre, but not the other way around. The solution:

get inverse offset

```
# assume this is the offset of the sensor from the robot centre
offset="1.3,-0.6,0.3,-0.5,0.6,-0.7"

inversed_offset=$( echo "0,0,0,0,0,0" | points-frame --to="$offset" --fields="x,y,z,roll,pitch,yaw" --
precision=16 )
```

Now `inversed_offset` is the position (and pose) of the robot centre in the coordinate system associated with the sensor.

Step by step demo:

- Start with some coordinates (navigation data in the world frame; the specific coordinate system does not matter). A sample data file is attached to this page:

  get nav

  ```
  cat nav.bin | csv-from-bin t,6d | head -n 2
  20101209T032242.279632,6248546.875440197,332966.0202201727,-37.76910082437098,-0.02045280858874321,
  -0.01195328310132027,2.113722085952759
  20101209T032242.280425,6248546.875484069,332966.020129519,-37.76909669768065,-0.02046919427812099,
  -0.0119620431214571,2.113716840744019
  ```

  The example uses binary, but this is up to you. The `nav.bin` file contains the trajectory of the robot centre (GPS unit) in the world frame.
- Get the coordinates of the sensor in the world frame:

  get sensor trajectory

  ```
  cat nav.bin | csv-paste "-;binary=t,6d" "value=$offset;binary=6d" \
      | points-frame --from --fields=",frame,x,y,z,roll,pitch,yaw" --binary="t,6d,6d" \
      | csv-shuffle --fields="t,,,,,,,,,,,,,x,y,z,roll,pitch,yaw" --binary="t,6d,6d,6d" --output-fields="t,
  x,y,z,roll,pitch,yaw" > sensor.bin
  ```

  Now `sensor.bin` is the trajectory of the sensor in the world frame. We want to get the trajectory of the robot centre from these data.
- Just do it:

  get centre coordinates back

  ```
  cat sensor.bin | csv-paste "-;binary=t,6d" "value=$inversed_offset;binary=6d" \
      | points-frame --from --fields=",frame,x,y,z,roll,pitch,yaw" --binary="t,6d,6d" \
      | csv-shuffle --fields="t,,,,,,,,,,,,,x,y,z,roll,pitch,yaw" --binary="t,6d,6d,6d" --output-fields="t,
  x,y,z,roll,pitch,yaw" > restored.bin
  ```

  Note the use of `inversed_offset`.
- Verify by comparing to the original `nav.bin`:

verify

```
cat nav.bin \
    | csv-paste "-;binary=t,6d" "restored.bin;binary=t,6d" \
    | csv-eval --full-xpath --binary="t,6d,t,6d" --fields="f/t,f/x,f/y,f/z,f/roll,f/pitch,f/yaw,s/t,s/x,s
/y,s/z,s/roll,s/pitch,s/yaw" \
        "dx = abs(f_x - s_x); dy = abs(f_y - s_y); dz = abs(f_z - s_z); droll = abs(f_roll - s_roll);
dpitch = abs(f_pitch - s_pitch); dyaw = abs(f_yaw - s_yaw);" \
    | csv-shuffle --fields=",,,,,,,,,,,,,,dx,dy,dz,droll,dpitch,dyaw" --binary="t,6d,t,6d,6d" --output-
fields="dx,dy,dz,droll,dpitch,dyaw" \
    | csv-calc --fields="dx,dy,dz,droll,dpitch,dyaw" --binary="6d" mean \
    | csv-from-bin 6d
```

The output is on the order of Unknown macro: 'mathinline'. The precision is defined by the accuracy of `inversed_offset` calculations above. If the `--precision=16` option were not given, the comparison would be valid up to Unknown macro: 'mathinline' or so.